ZZW
AF

Docket No.: 049657-0921 PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of	:	Customer Number: 20277
Toyohiko YOSHIDA	:	Confirmation Number: 6032
Application No.: 09/756,863	:	Tech Center Art Unit: 2183
Filed: January 10, 2001	:	Examiner: Aimee J. Li

For: INSTRUCTION TRANSLATOR TRANSLATING NON-NATIVE INSTRUCTIONS FOR A PROCESSOR INTO NATIVE INSTRUCTIONS THEREFOR, INSTRUCTION MEMORY WITH SUCH TRANSLATOR, AND DATA PROCESSING APPARATUS USING THEM

TRANSMITTAL OF SUBSTITUTE APPEAL BRIEF

Mail Stop Appeal Brief
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

Submitted herewith is Appellant's Substitute Appeal Brief in support of the Notification of Non-Compliant Appeal Brief issued July 27, 2006.

To the extent necessary, a petition for an extension of time under 37 C.F.R. 1.136 is hereby made. Please charge any shortage in fees due under 37 C.F.R. 1.17 and 41.20, and in connection with the filing of this paper, including extension of time fees, to Deposit Account 500417 and please credit any excess fees to such deposit account.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP

Tomoki Tanida
Limited Recognition No. L0098

600 13th Street, N.W.
Washington, DC 20005-3096
Phone: 202.756.8000 SAB:TT
Facsimile: 202.756.8087
Date: December 11, 2006

Please recognize our Customer No. 20277 as our correspondence address.

TABLE OF CONTENTS

Page

Docket No.: 0490-7-0921 PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of	:	Customer Number: 20277
Toyohiko YOSHIDA	:	Confirmation Number: 6032
Application No.: 09/756,863	:	Tech Center Art Unit: 2183
Filed: January 10, 2001	:	Examiner: Aimee J. Li

For: INSTRUCTION TRANSLATOR TRANSLATING NON-NATIVE INSTRUCTIONS FOR A PROCESSOR INTO NATIVE INSTRUCTIONS THEREFOR, INSTRUCTION MEMORY WITH SUCH TRANSLATOR, AND DATA PROCESSING APPARATUS USING THEM

SUBSTITUTE APPEAL BRIEF

Mail Stop Appeal Brief
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed February 28, 2006, wherein Appellant appeals from the Primary Examiner's rejection of claims 1-12, 14, 16, 18 and 19.

I. Real Party In Interest

This application is assigned to Renesas Technology Corp. by assignment recorded on September 10, 2003, at Reel 014502, Frame 0282.

II. Related Appeals and Interferences

Appellant is unaware of any related Appeal or Interference.

III. Status of Claims

Claims 1-12, 14, 16, 18 and 19, all pending claims, have been finally rejected. It is from the final rejection of claims 1-12, 14, 16, 18 and 19 that this Appeal is taken.

IV. Status of Amendments

No amendment has been filed subsequent to the imposition of the rejection in the Official Action issued December 2, 2005.

V. Summary of Claimed Subject Matter

The claims are directed to an instruction translator, an instruction memory attached with a translator and a data processing apparatus. A data processing apparatus is exemplarily illustrated in Fig. 2, and described in the specification beginning at page 11, line 1. The disclosed data processing apparatus includes processor 10 including processor core, having a VLIW (Very Long Instruction Word) type instruction architecture. See, page 11, line 3; page 11, line 20 to page 12, line 1; and Fig. 3. The data processing apparatus also comprises an instruction memory with a translator which stores an instruction in an architecture different from that of processor 10. For example, the disclosed data processing apparatus comprises memory 24 with translator 14 for compressed instruction, memory 25 with translator 15 for JAVA instruction, and memory 26 with translator 16 for non-native instruction X. See, page 11, lines 4-8. The data processing apparatus further includes RAM 21 for native instruction, data memory 22, and ROM 23 to store a native instruction, a compressed instruction, a JAVA instruction, a non-native instruction X and data. See, page 11, lines 8-11. All the above elements are coupled to bus 40, and communication through bus 40 is controlled by bus control circuit 20. See, page 11, line 3.

Bus control circuit 20 decodes an address output from processor 10 onto bus 40, and outputs a chip select signal CS, which activates one of RAM 21 for native instructions, data memory 22, ROM 23, and memories 24-26. Bus control circuit 20 also outputs a translation function enable signal TE to translators 14-16 to control instruction translation functions of the translators. See, page 11, lines 12-21.

Fig. 12 exemplarily shows memory 24 includes translator 14 and RAM 245 for compressed instruction as an instruction memory attached with a translator. See, page 20, lines 9-10. Translator 14 includes translation circuit 243 including an instruction code expansion portion 350 which expands two instructions input from RAM 245 through instruction code output line 257 into an 8-byte VLIW instruction, and outputs it onto an output line 360. See page 22, lines 1-4; and Fig. 13.

Translator 14 also includes cache memory 354. If the cache memory holds an expanded instruction corresponding to an address applied from address line 254, the memory outputs an expanded VLIW instruction onto an output line 359 and asserts a cache hit signal on cache hit signal line 357. See, page 21, line 30 to page 22, line 1; and Fig. 13.

Furthermore, translator 14 includes MUX 356 which selects either one of output line 359 from cache memory 354 or output line 360 from instruction code expansion portion 350, and outputs the selected signal. See, page 22, lines 4-7.

In operation, an address input from address line 254 is input to cache memory 354 and instruction expansion portion 350. Cache memory 354 checks whether or not an instruction of the same address as that of the input address has been expanded in the past and held therein. If such an instruction is held, an expanded VLIW instruction is output to output line 359 from cache memory 354, and a cache hit signal on cache hit signal line 357 is asserted, which indicates that cache memory 354 has a cache hit. At this time, MUX 356 selects the output on output line 359 and outputs the

expanded VLIW instruction output from cache memory onto data line 259. See, page 22, lines 13-25, and Fig. 13.

When cache memory 354 has a miss, an expanded 8-byte VLIW instruction output from instruction code expansion portion 350 to output line 360 is selected by MUX 356, and is output to data line 259. At this time, the expanded instruction code output onto data line 259 is transferred to cache memory 354 as well, and written into a corresponding entry. See, page 22, line 26 to page 23, line 1, and Fig. 13.

Independent claim 1 recites an instruction translator exemplified by Figs. 2, 12, 13, 22, and 23. In a processor (10 in Figs. 2 and 3) operating with instructions in a first instruction architecture (VLIW type instruction architecture) as a native instruction, an instruction translator (14-16 in Fig. 2) used with an instruction memory (24-26 in Fig. 2) to store an instruction in a second instruction architecture (compressed instruction, JAVA instruction, and non-native instruction X) different from said first instruction architecture, for translating an instruction in said second instruction architecture into an instruction in said first instruction architecture for execution by said processor (10 in Fig. 2) in said first instruction architecture (see page 11, lines 1-11; and page 11, line 22 to page 12, line 1), said instruction translator (243 in Fig. 12, and 263 in Fig. 22) comprising:

a translator (350 in Fig. 13 and 370 in Fig. 23) for reading out an instruction from said instruction memory (245 in Fig. 12 and 265 in Fig. 22) in response to a received first address (ADDRESS on address line 254 in Figs. 12 and address line 274 in Fig. 22) in said instruction memory (245 in Fig. 12 and 265 in Fig. 22) of an instruction to be executed by said processor (10 in Fig. 2) and translating the read out instruction in said second instruction architecture into an instruction in said first instruction architecture, wherein said processor (10 in Fig. 2) is configured to execute instructions only in said first instruction architecture (see page 22, lines 1-4 and page 28, lines 12-16);

an instruction cache (354 in Fig. 13 and 374 in Fig. 23) for temporarily holding the instruction in said first instruction architecture after the translation by said translator (350 in Fig. 13 and 370 in Fig. 23) in association with the first address in said instruction memory (see page 22, line 13 to page 23, line 1); and

a selector (356 in Fig. 13 and 376 in Fig. 23) for searching said instruction cache (354 in Fig. 13 and 374 in Fig. 23) in response to a received second address (ADDRESS on address line 254 in Fig. 12 and address line 274 in Fig. 22) of an instruction to be executed by said processor (10 in Fig. 1), and for selectively outputting, based on a determination result of whether or not an instruction corresponding to the instruction of the second address is held in said instruction cache (354 in Fig. 13 and 374 in Fig. 13), an instruction output by said translator (350 in Fig. 13 and 370 in Fig. 23) or the corresponding instruction held in said instruction cache (354 in Fig. 13 and 374 in Fig. 23) (see page 22, lines 4-7; page 22, line 13 to page 23, line 1; page 27, lines 18-20; and page 28, lines 19-32).

Independent claim 12 recites an instruction memory attached with a translator exemplified by Figs. 2, 12, 13, 22 and 23. The instruction memory (24-26 in Fig. 2) attached with the translator (14-16 in Fig. 2), used with a processor (10 in Figs. 2 and 3) configured to execute instructions in a first instruction architecture (VLIW type instruction architecture) as a native instruction (see page 11, lines 1-11; and page 11, line 22 to page 12, line 1), comprising:

an instruction storage unit (245 in Fig. 12 and 265 in Fig. 22) to store an instruction in a second instruction architecture (e.g., compressed instruction and JAVA instruction) (see page 20, lines 9-10 and page 25, lines 23-24); and

an instruction translator (243 in Fig. 12 and 263 in Fig. 22) to translate an instruction in said second instruction architecture output from said instruction storage unit (245 in Fig. 12 and 265 in Fig. 22) into an instruction in said first instruction architecture for application to said processor (10 in Fig.

2), wherein said processor (10 in Fig. 2) is configured to execute instructions only in said first instruction architecture;

said instruction translator (243 in Fig. 12 and 263 in Fig. 22) including:

a translator (350 in Fig. 13 and 370 in Fig. 23) to read out the instruction in said second instruction architecture from said instruction storage unit (245 in Fig. 12 and 265 in Fig. 22) in response to a received first address (ADDRESS on address line 254 in Fig. 22 and address line 274 in Fig. 22) of an instruction to be executed by said processor (10 in Fig. 2) and translate the read out instruction in said second instruction architecture into the instruction in said first instruction architecture (see page 22, lines 1-4 and page 28, lines 12-16);

an instruction cache (354 in Fig. 13 and 374 in Fig. 23) to temporarily hold the instruction in said first instruction architecture after the translation by said translator (350 in Fig. 13 and 370 in Fig. 13) in association with the first address (see page 22, line 13 to page 23, line 1); and

a selector (356 in Fig. 13 and 376 in Fig. 23) to search said instruction cache (354 in Fig. 13 and 374 in Fig. 23) in response to a received second address (ADDRESS on address line 254 in Fig. 22 and address line 274 in Fig. 22) of an instruction to be executed by said processor (10 in Fig. 2) and selectively output to said processor (10 in Fig. 2), based on a determination result of whether or not an instruction corresponding to the instruction of the address is held in said instruction cache (354 in Fig. 13 and 374 in Fig. 23), an instruction output by said translator (350 in Fig. 13 and 370 in Fig. 23) or the corresponding instruction in said first instruction architecture held in said instruction cache (354 in Fig. 13 and 374 in Fig. 23) (see page 22, lines 4-7; page 22, line 13 to page 23, line 1; page 27, lines 18-20; and page 28, lines 19-32).

Independent claim 16 recites a data processing apparatus exemplified by Figs. 2, 12, 13, 22 and 23. The data processing apparatus comprising:

a processor (10 in Figs. 2 and 3) operating with an instruction in a first instruction architecture (VLIW type instruction architecture) as a native instruction (see page 11, line 28 to page 12, line 1);

a bus (40 in Fig. 2) to which said processor (10 in Fig. 2) is connected (see page 11, lines 12-21);

a first instruction memory (24-26 in Fig. 2) with a translator (14-16 in Fig. 2) interconnected with said processor (10 in Fig. 2) through said bus (40 in Fig. 2) (see page 11, lines 4-11); and

a second instruction memory (21-23 in Fig. 2) interconnected to said processor (10 in Fig. 2) through said bus (40 in Fig. 2) (see page 11, lines 4-11),

said first instruction memory (24-26 in Fig. 2) with a translator (14-16 in Fig. 2) including:

a first instruction storage unit (245 in Fig. 12 and 265 in Fig. 22) to store an instruction in a second instruction architecture (e.g., compressed instruction and JAVA instruction) transferred from said processor (10 in Fig. 2) through said bus (40 in Fig. 2) (see page 19, line 24 to page 20, line 8; and Fig. 11); and

an instruction translator (243 in Fig. 12 and 263 in Fig. 22) to translate the instruction in said second instruction architecture output from said first instruction storage unit (245 in Fig. 12 and 265 in Fig. 22) into an instruction in said first instruction architecture for application to said processor (10 in Fig. 1) through said bus (40 in Fig. 2) (see pages 22, lines 1-4 and page 28, lines 12-16, and see, also, page 8, lines 13-29), and

said second instruction memory (21-23 in Fig. 2) including:

a second instruction storage unit (21-23 in Fig. 2) to store an instruction in said first instruction architecture transferred from said processor (10 in Fig. 2) through said bus (40 in Fig. 2) (see page 19, line 24 to page 20, line 8; and Fig. 11); and

an instruction reading circuit (21-23 in Fig. 2) responsive to an address signal applied from said processor (10 in Fig. 1) through said bus for applying an instruction in said first instruction architecture output from said second instruction storage unit to said processor through said bus (40 in Fig. 2) (see, also, page 8, line 30 to page 9, line 9).

VI. Grounds of Rejection To Be Reviewed By Appeal

A. Claims 1, 5, 12, 16, 18 and 19 stand rejected as being unpatentable over U.S. Patent 5,638,525 ("Hammond") in view of IBM Technical Disclosure Bulletin, *Instruction Cache Bypass During Cache Reload* ("IBM") under 35 U.S.C. §103(a).

B. Claims 2 and 3 stand rejected as being unpatentable over Hammond in view of IBM, and further in view of U.S. Patent 5,875,336 ("Dickol") under 35 U.S.C. §103(a).

C. Claim 4 stands rejected as being unpatentable over Hammond in view of IBM, in view of Dickol, and further in view of U.S. Patent 5,313,614 ("Goettelmann") under 35 U.S.C. §103(a).

D. Claims 6 and 7 stand rejected as being unpatentable over Hammond, in view of IBM, and further in view of Goettelmann under 35 U.S.C. §103(a).

E. Claims 8 and 10 stand rejected as being unpatentable over Hammond, in view of IBM, and further in view of U.S. Patent 5,023,776 ("Gregor") under 35 U.S.C. §103(a).

F. Claims 9 and 11 stand rejected as being unpatentable over Hammond, in view of IBM, in view of Gregor, and further in view of U.K. Patent Application GB220481A ("Schacham") under 35 U.S.C. §103(a).

G. Claim 14 stands rejected as being unpatentable over Hammond, in view of IBM, and further in view of U.S. Patent 5,953,520 ("Mallick") under 35 U.S.C. §103(a).

VII. Argument

A. Rejection of claims 1, 5, 12, 16, 18 and 19 under 35 U.S.C. §103(a) over Hammond and IBM

Legal precedent is well developed on the subject of obviousness in the application of a rejection under 35 U.S.C. §103. It is incumbent upon the examiner to factually support a conclusion of obviousness. *In re Mayne,* 104 F.3d 1339, 41 USPQ2d 1451 (Fed. Cir. 1997*); In re Oetiker*, 977 F.2d 1443, 24 USPQ2d 1443 (Fed. Cir. 1992). The examiner must provide a reason why one having ordinary skill in the art would have been led to modify a particular prior art reference in a particular manner to arrive at a particular claimed invention; *Ecolochem Inc. v. Southern California Edison, Co.* 227 F.3d 361, 56 USPQ2d 1065 (Fed. Cir. 2000); *In re Rouffet*, 149 F.3d 1350, 47 USPQ2d 1453 (Fed. Cir. 1998). *Ashland Oil, Inc. v. Delta Resins & Refractories, Inc.*, 776 F.2d 281, 227 USPQ 657 (Fed. Cir. 1985); *In re Fine*, 837 F.2d 1071, 5 USPQ2d 1596 (Fed. Cir. 1988); *Stratoflex, Inc. v. Aeroquip Corp.*, 713 F.2d 1530, 218 USPQ 871 (Fed. Cir. 1983); *In re Warner*, 379 F.2d 1011, 154 USPQ 173 (CCPA 1967).

In order to establish the requisite motivation, "clear and particular" factual findings must be made as to a specific understanding or specific technological principle which would have realistically compelled one having ordinary skill in the art to modify a particular reference to arrive at the claimed invention based upon facts-- not generalizations. *Ruiz v. A.B. Chance Co.*, 234 F.3d 654, 57 UPSQ2d 1161 (Fed. Cir. 2000); *Ecolochem Inc. v. Southern California Edison, Co.* 227 F.3d 361, 56 USPQ2d 1065 (Fed. Cir. 2000); *In re Kotzab*, 217 F.3d 1365, 55 USPQ 1313 (Fed. Cir. 2000); *In re Dembiczak*, 175 F.3d 994, 50 USPQ2d 1614 (Fed. Cir. 1999). What may or may not be known in general does not establish the requisite realistic motivation for obviousness; see *In re Deuel*, 51 F.3d 1552, 34 USPQ2d

1210 (Fed. Cir. 1995). Moreover, merely identifying features of a claimed invention in disparate prior art references does not, automatically, establish the requisite motivation for combining references in any particular manner. *In re Dembiczak*, 175 F.3d 994, 50 USPQ2d 1614 (Fed. Cir. 1999); *Grain Processing Corp. v. American-Maize Products Co.*, 840 F.2d 902, 5 USPQ2d 1788 (Fed. Cir. 1988).

1. Claim 1

Claim 1 is an independent claim and is reproduced as follows:

> 1. In a processor operating with instructions in a first instruction architecture as a native instruction, an instruction translator used with an instruction memory to store an instruction in a second instruction architecture different from said first instruction architecture, for translating an instruction in said second instruction architecture into an instruction in said first instruction architecture for execution by said processor in said first instruction architecture said instruction translator comprising:
>
> a translator for reading out an instruction from said instruction memory in response to a received first address in said instruction memory of an instruction to be executed by said processor and translating the read out instruction in said second instruction architecture into an instruction in said first instruction architecture, wherein said processor is configured to execute instructions only in said first instruction architecture;
>
> an instruction cache for temporarily holding the instruction in said first instruction architecture after the translation by said translator in association with the first address in said instruction memory; and
>
> a selector for searching said instruction cache in response to a received second address of an instruction to be executed by said processor, and for selectively outputting, based on a determination result of whether or not an instruction corresponding to the instruction of the second address is held in said instruction cache, an instruction output by said translator or the corresponding instruction held in said instruction cache.

The translator receives a non-native architecture instruction and translates it into one or more corresponding translated instructions. An instruction cache is provided for temporarily storing the translated instruction(s). A selector provides an instruction to the processor for execution. The processor is configured to execute only native format instructions. The selector determines whether to

provide a translated instruction from the instruction cache or a translated instruction from the translator.

1-1. Examiner's arguments in paragraphs 5 and 6 of the Office Action

Hammond, a primary reference, discloses a processor system intended to execute an instruction set even if the instruction set is any of a first instruction set and a second instruction set. The first instruction set is an instruction set different from the second instruction set. In Fig. 5 showing a processor, both of the first instruction set and the second instruction set are stored in an external memory. In the first instruction set mode, a translator operates to translate the first instruction set into the second instruction set, and in the second instruction set mode, the instructions of the second instruction set are transferred directly to the processor from the external memory without passing through the translator so that the instructions are executed by an execution unit. The instructions of the second instruction set are decoded so as to be executed by the execution unit in both the first instruction set mode and the second instruction set mode. A demultiplexer is used to switch an instruction transfer path in accordance with a signal produced by the decoder decoding a switch instruction (jmpx instruction) between the first instruction set mode and the second instruction set mode. In Fig. 5 of Hammond, the instructions read from the external memory or the translated instructions are normally stored in an instruction cache.

In paragraph 6 of the Office Acton, the Examiner admitted that Hammond, described above, does not explicitly teach the "selector for searching said instruction cache in response to a received second address of an instruction to be executed by said processor, and for selectively outputting, based on a determination result of whether or not an instruction corresponding to the instruction of the second address is held in said instruction cache, an instruction output by said translator or the

corresponding instruction held in said instruction cache," recited in claim 1. However, the Office Action applied IBM, and asserted that IBM teaches the selection of either an instruction stored in an instruction cache or an instruction from the main memory based on whether there is a cache miss so as to reduce the delay associated with waiting for an instruction while a cache line is being retrieved on a cache miss.

The Office Action concluded that one of ordinary skill in the art would have found it obvious to modify Hammond's processor to include a cache bypass around the instruction cache of Hammond so that the instructions can be selected from either the output of the translator or the instruction cache based on a cache hit/miss in order to reduce the delay associated with a fetch to memory on an instruction cache miss. However, Hammond's processor does not have any delay issue associated with a cache miss.

Hammond discloses as follows:

> In this example, processor 104 powers up in the first instruction set mode. Thus demultiplexer 540 initially receives a first signal from decoder 543 and selects translator 541. Instructions transmitted from external memory through demultiplexer 540 is translated into instructions of the second instructions set by translator 541. Translator 541 comprises circuitry for.... The translated instructions are then transmitted to instruction cache 542. Instruction cache 542 stores the instructions until they are decoded and executed. At the appropriate time for decoding instructions, instructions in instruction cache 542 are transmitted to decoder 543 for decoding of instructions from instruction cache 542. The instructions from instruction cache 542 are instructions of the second instruction set. After decoding, instructions of the second instruction set are transmitted to execution unit 544.

Column 15, lines 12-29 of Hammond (emphasis added).

It is apparent that instruction cache 542 of Hammond is equivalent to a buffer, or an instruction memory because the instructions in the instruction cache are sequentially executed according to Hammond's disclosure. Moreover, Hammond fails to teach retrieving an instruction from the instruction cache in accordance with an address from the execution unit, and determining whether an

addressed instruction is cached in the instruction cache. Hammond does not require the caching of instructions and determination of cache miss/hit in accordance with an address. This instruction cache is, thus, not associated with cache miss penalty. Since there is no cache miss in Hammond, it is not necessary for Hammond's processor to have a cache bypass around the instruction cache of Hammond to reduce the delay associated with a cache miss. Avoiding any delay associated with a cache miss does not impel one having ordinary skilled in the art to modify Hammond's processor to have a cache bypass because Hammond does not have any cache miss issue.

1-2. The Examiner's arguments in paragraphs 32 and 33 of the Office Action

The Examiner further asserted as follows:

> Applicants' arguments are based upon whether or not Hammond has taught cache miss within their device. However, that is exactly why IBM is relied upon. Hammond was relied upon to teach the other aspect of the claimed subject matter.

This assertion does not address the issue of why an artisan would have been motivated to modify the prior art. The Examiner simply identified features of the claimed invention as being disclosed in either of the two prior art references, which does not automatically establish the requisite motivation for combining references in any particular manner. *In re Dembiczak*, and *Grain Processing Corp. v. American-Maize Products Co., supra.*

The Examiner continued to assert as follows:

> However, to say that a cache miss is impossible within Hammond is false. A cache miss is possible in any system that uses caches or buffering systems. Caches and buffers are limited in space, so it cannot always be guaranteed that the next instruction is within the cache, especially when there is a branch instruction executing. The branch target instruction and the rest of the instructions along the taken path are not guaranteed to be in the cache. If these instructions are not in the cache, then the instructions needs to be fetched from main memory and the cache needs to be reloaded with these instructions. IBM has taught a method to reload the cache and a bypass to the instruction cache so that the processor does not need to necessarily wait for the cache to be reloaded, e.g., avoid idle cycles while the cache is filled with the correct instruction path.

The Examiner discussed a cache miss as a general matter. However, the issue here is whether there is a cache miss issue in Hammond, i.e., whether Hammond requires caching instructions, and determining cache miss/hit. As discussed above, Hammond fails to teach retrieving an instruction from the instruction cache in accordance with an address from the execution unit, and determining whether an addressed instruction is cached in the instruction cache. Appellant submits that the above discussion regarding cache miss by the Examiner is based on hindsight consideration of the teachings of the present invention.

The Examiner further asserted that "[t]he branch target instruction and the rest of the instructions along the taken path are not guaranteed to be in the cache. If these instructions are not in the cache, then the instructions need to be fetched from main memory and the cache needs to be reloaded with these instructions" (paragraph 33 of the Office Action, cited above).

Upon execution of a branch instruction, various schemes are employed for avoiding delay required until a branch target is determined. In one scheme such as a branch delay scheme, an instruction such as a NOP instruction is inserted after the branch instruction. In another scheme, a branch instruction is executed based on branch prediction, in which a branch taken or not taken by the branch instruction is predicted, and a next instruction is executed. If the prediction is incorrect, the next instruction under execution is invalidated and a correct branch target instruction and subsequent instructions are fetched into an instruction memory.

The above-described branch prediction miss is different from the cache miss in which an access-required instruction is not present in the cache. In the branch prediction miss, the addressed instruction is present in the instruction cache, but the addressed instruction is not a correct one to be executed subsequently. Thus, the branch prediction miss is completely different from the cache miss, contrary to the Examiner's assertion.

Based upon the foregoing, Appellant submits that the Examiner has not established a *prima facie* basis to deny patentability to the claimed invention for lack of the requisite realistic motivation. The applied combination of Hammond and IBM would not have led an artisan to a modification that would result in the claimed invention.

2. Claim 12

Claim 12 is an independent claim and is reproduced as follows:

> 12. An instruction memory attached with a translator, used with a processor configured to execute instructions in a first instruction architecture as a native instruction, comprising:
>
> an instruction storage unit to store an instruction in a second instruction architecture; and
>
> an instruction translator to translate an instruction in said second instruction architecture output from said instruction storage unit into an instruction in said first instruction architecture for application to said processor, wherein said processor is configured to execute instructions only in said first instruction architecture;
>
> said instruction translator including:
>
> a translator to read out the instruction in said second instruction architecture from said instruction storage unit in response to a received first address of an instruction to be executed by said processor and translate the read out instruction in said second instruction architecture into the instruction in said first instruction architecture;
>
> an instruction cache to temporarily hold the instruction in said first instruction architecture after the translation by said translator in association with the first address; and
>
> a selector to search said instruction cache in response to a received second address of an instruction to be executed by said processor and selectively output to said processor, based on a determination result of whether or not an instruction corresponding to the instruction of the address is held in said instruction cache, an instruction output by said translator or the corresponding instruction in said first instruction architecture held in said instruction cache.

In paragraph 10 of the Office Action, the Examiner recognized that Hammond does not teach the "selector," but asserted that IBM teaches the missing feature of Hammond. The Examiner's

asserted motivation to modify Hammond based on IBM to arrive at the claimed invention is that one of ordinary skill in the art would have found it obvious to modify the processor of Hammond to include a cache bypass around the instruction cache of Hammond so that instructions can be selected from either the output of the translator or the instruction cache based on a cache hit/miss in order to reduce the delay associated with a fetch to memory on an instruction cache miss.

Appellant submits that instruction cache 542 of Hammond is equivalent to a buffer, or an instruction memory. Therefore, since there is no cache miss in Hammond, it is not necessary to modify Hammond to have a cache bypass. The discussion above with respect to the rejection of claim 1 is applicable to the rejection of claim 12.

Accordingly, it is submitted that the Examiner has not established a *prima facie* basis to deny patentability to the claimed invention for lack of the requisite realistic motivation. The applied combination of Hammond and IBM would not have led an artisan to a modification that would result in the claimed invention.

3. Claim 16

Claim 16 is an independent claim and is reproduced as follows:

16. A data processing apparatus, comprising:

a processor operating with an instruction in a first instruction architecture as a native instruction;

a bus to which said processor is connected;

a first instruction memory with a translator interconnected with said processor through said bus; and

a second instruction memory interconnected to said processor through said bus,

said first instruction memory with a translator including:

a first instruction storage unit to store an instruction in a second instruction architecture transferred from said processor through said bus; and

an instruction translator to translate the instruction in said second instruction architecture output from said first instruction storage unit into an instruction in said first instruction architecture for application to said processor through said bus, and

said second instruction memory including:

a second instruction storage unit to store an instruction in said first instruction architecture transferred from said processor through said bus; and

an instruction reading circuit responsive to an address signal applied from said processor through said bus for applying an instruction in said first instruction architecture output from said second instruction storage unit to said processor through said bus.

It is submitted that the applied combination of Hammond et al. and IBM does not teach a data processing apparatus including all the limitations recited in claim 16.

In paragraph 11 of the Office Action, the Examiner asserted that the claimed first and second instruction storage units are disclosed in Hammond et al. However, the Examiner overlooked the following limitations: "store an instruction in a second instruction architecture transferred from said processor through said bus" for the first instruction storage unit, and "store an instruction in said first instruction architecture transferred from said processor through said bus" for the second instruction storage unit (emphasis added). Hammond et al. discloses one memory for storing instructions of the first instruction set and instructions of the second instruction set. Hammond et al. does not teach transferring instructions from a processor to the memory through a bus.

Furthermore, in paragraph 12 of the Office Action, the Examiner recognized that Hammond does not teach the "instruction reading circuit," but asserted that IBM teaches the missing feature of Hammond. IBM purportedly teaches the selection of either an instruction stored in an instruction cache or an instruction from the main memory based on if there is a cache miss, according to the Examiner.

In contrast, the instruction reading circuit of claim 16 is configured to be responsive to an address signal applied from said processor through said bus for applying an instruction in said first instruction architecture output from said second instruction storage unit to said processor through said bus. What is disclosed by IBM, i.e., a cache miss, is irrelevant to the claimed instruction reading circuit. Claim 16 does not require selection of either an instruction stored in an instruction cache or an instruction from a memory based on whether there is a cache miss. Claim 16 requires the instruction reading circuit to read out an instruction in the first instruction architecture from the second instruction storage unit and to apply it to the processor through the bus.

Therefore, the applied combination of Hammond and IBM, does not teach each and every limitation recited in independent claim 16.

Appellant further submits that there is no motivation to modify Hammond's processor based on the teachings of IBM to arrive at the claimed invention. The Examiner's asserted motivation to modify Hammond based on IBM to arrive at the claimed invention is that one of ordinary skill in the art would have found it obvious to modify the processor of Hammond to include a cache bypass around the instructions cache of Hammond so that instructions can be selected from either the output of the translator or the instruction cache based on a cache hit/miss in order to reduce the delay associated with a fetch to memory on an instruction cache miss.

Appellant submits that instruction cache 542 of Hammond is equivalent to a buffer, or an instruction memory. Therefore, since there is no cache miss in Hammond, it is not necessary to modify Hammond to have a cache bypass. The discussion above with respect to the rejection of claim 1 is applicable to the rejection of claim 16 in this aspect.

Accordingly, the Examiner has not established a prima facie basis to deny patentability to the claimed invention for lack of the requisite realistic motivation. The applied combination of Hammond and IBM would not have led an artisan to a modification that would result in the claimed invention.

4. Claim 5

Claim 5 is dependent from claim 1 and additionally requires the following:

said translator includes a plurality of translators which translate a plurality of instructions in said second instruction architecture read out from said instruction memory into one instruction in said first instruction architecture.

The Examiner asserted, "[w]hile a plurality of translators is not explicitly taught, the translation of a plurality of instructions in the second instruction architecture into one instruction in the first instruction architecture is taught" (paragraph 8 of the Office Action). However, the Examiner did not provide any reason to support his position. Appellant submits that neither Hammond nor IBM teaches the additional requirements.

Accordingly, it is urged that the Examiner has not established a prima facie basis to deny patentability to the claimed invention. The applied combination of Hammond and IBM would not have led an artisan to a modification that would result in the claimed invention.

5. Claim 18

Claim 18 is dependent from claim 16 and additionally requires the following:

the number of clock cycles spent by said processor to access said first instruction memory through said bus is larger than the number of clock cycles spent by said processor to access said second instruction memory through said bus.

Appellant submits that neither Hammond nor IBM teaches the additional requirements. The Examiner asserted that IBM teaches the additional feature, but the Examiner's cited portion appears to be irrelevant to claim 18.

Accordingly, it is urged that the Examiner has not established a prima facie basis to deny patentability to the claimed invention. The applied combination of Hammond and IBM would not have led an artisan to a modification that would result in the claimed invention.

6. Claim 19

Claim 19 is dependent from claim 16 and additionally requires the following:

> The data processing apparatus according to claim 16, further comprising a third instruction memory with a translator interconnected to said processor through said bus,
>
> said third instruction memory with a translator including:
>
> a third instruction storage unit to store an instruction in a third instruction architecture different from said second instruction architecture, said instruction in said third instruction architecture being transferred from said processor through said bus; and
>
> an instruction translation circuit responsive to an address signal applied from said processor through said bus, for translating an instruction in said third instruction architecture output from said third instruction storage unit into an instruction in said first instruction architecture for application to said processor.

Appellant submits that neither Hammond nor IBM teaches the additional requirements. At least, the Examiner overlooked the following requirement: "said instruction in said third instruction architecture being transferred from said processor through said bus," as discussed with respect to the rejection of parent claim 16 (emphasis added).

Accordingly, it is urged that the Examiner has not established a prima facie basis to deny patentability to the claimed invention. The applied combination of Hammond and IBM would not have led an artisan to a modification that would result in the claimed invention.

B. Rejection of claims 2 and 3 under 35 U.S.C. §103(a) over Hammond, IBM and Dickol

1. Claim 2

Claim 2 is dependent from claim 1 and additionally requires the following:

said second instruction architecture is a variable length instruction architecture, and said translator includes a variable length translator for translating an instruction in said second instruction architecture read out from said instruction memory into one or more instructions in said first instruction architecture, the number of which depends on an instruction length of the read out instruction in said second instruction architecture.

The Examiner applied Dickol for teaching the additional requirements in claim 2. However, Appellant submits that Dickol does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 2 is patentably distinguishable over Hammond, IBM and Dickol at least because the claim includes all the limitations recited in independent claim 1. The applied combination of Hammond, IBM and Dickol would not have led an artisan to a modification that would result in the claimed invention.

2. Claim 3

Claim 3 is dependent from claim 1 and additionally requires the following:

said variable length translator translates the instruction in said second instruction architecture read out from said instruction memory into an instruction in said first instruction architecture having a total length depending on and larger than the instruction length of said read out instruction in said second instruction architecture.

The Examiner applied Dickol for teaching the additional requirements in claim 3. However, Appellant submits that Dickol does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 3 is patentably distinguishable over Hammond, IBM and Dickol at least because the claim includes all the limitations recited in claims 1 and 2. The applied combination of Hammond, IBM and Dickol would not have led an artisan to a modification that would result in the claimed invention.

C. Rejection of claim 4 under 35 U.S.C. §103(a) over Hammond, IBM, Dickol and Goettelmann

Claim 4 is dependent from claim 1 and additionally requires the following:

each instruction in said first instruction architecture includes one or a plurality of sub instructions, and the number of the sub instructions included in the instruction in the first instruction architecture translated by said variable length translator depends on the instruction length of said read out instruction in said second instruction architecture.

The Examiner applied Goettelmann for teaching the additional requirements in claim 4. However, Appellant submits that Goettelmann does not cure the deficiencies of the applied combination of Hammond, IBM and Dickol. Accordingly, claim 4 is patentably distinguishable over Hammond, IBM, Dickol and Goettelmann at least because the claim includes all the limitations recited in claims 1-3. The applied combination of Hammond, IBM, Dickol and Goettelmann would not have led an artisan to a modification that would result in the claimed invention.

D. Rejection of claims 6 and 7 under 35 U.S.C. §103(a) over Hammond, IBM and Goettelmann

1. Claim 6

Claim 6 is dependent from claim 1 and additionally requires the following:

each instruction in said first instruction architecture can include one or a plurality of sub instructions, and

said translator translates a plurality of instructions in said second instruction architecture read out from said instruction memory into an instruction in said first instruction architecture including sub instructions, the number of which depends on the number of said plurality of instructions.

The Examiner applied Goettelmann for teaching the additional requirements in claim 6. However, Appellant submits that Goettelmann does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 6 is patentably distinguishable over Hammond, IBM and Goettelmann at least because the claim includes all the limitations recited in independent claim 1. The applied combination of Hammond, IBM and Goettelmann would not have led an artisan to a modification that would result in the claimed invention.

2. Claim 7

Claim 7 is dependent from claim 1 and additionally requires the following:

> the number of sub instructions included in the instruction in said first instruction architecture after said translation is equal to the number of said plurality of instructions.

The Examiner applied Goettelmann for teaching the additional requirements in claim 7. However, Appellant submits that Goettelmann does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 7 is patentably distinguishable over Hammond, IBM and Goettelmann at least because the claim includes all the limitations recited in independent claim 1. The applied combination of Hammond, IBM and Goettelmann would not have led an artisan to a modification that would result in the claimed invention.

E. Rejection of claims 8 and 10 under 35 U.S.C. §103(a) over Hammond, IBM and Gregor

1. Claim 8

Claim 8 is dependent from claim 1 and additionally requires the following:

> said translator translates said read out instruction in said second instruction architecture into one or a plurality of instructions as the instruction in said first instruction architecture;
>
> said instruction translator further comprises a controller for controlling said instruction cache so that said instruction cache holds each of said one or said plurality of instructions held in said instruction cache as an entry which can be invalidated in one of first and second conditions.

The Examiner applied Gregor for teaching the additional requirements in claim 8. However, Appellant submits that Gregor does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 8 is patentably distinguishable over Hammond, IBM and Gregor at least because the claim includes all the limitations recited in independent claim 1. The applied combination of Hammond, IBM and Gregor would not have led an artisan to a modification that would result in the claimed invention.

2. Claim 10

Claim 10 is dependent from claim 1 and additionally requires the following:

> said controller outputs a signal asserted when a new instruction cannot be held in said instruction cache without invalidating an entry which can be invalidated in the second condition.

The Examiner applied Gregor for teaching the additional requirements in claim 10. However, Appellant submits that Gregor does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 10 is patentably distinguishable over Hammond, IBM and Gregor at least because the claim includes all the limitations recited in independent claim 1. The applied combination of Hammond, IBM and Gregor would not have led an artisan to a modification that would result in the claimed invention.

F. Rejection of claims 9 and 11 under 35 U.S.C. §103(a) over Hammond, IBM, Gregor and Schacham

1. Claim 9

Claim 9 is dependent from claim 1 and additionally requires the following:

> said first condition is a holding control condition by hardware control based on a prescribed algorithm by said instruction cache, and
>
> said second condition is a condition in which an explicit invalidation instruction is applied from the outside of said instruction cache.

The Examiner further applied Schacham for teaching the additional requirements in claim 9. However, Appellant submits that Schacham does not cure the deficiencies of the applied combination of Hammond, IBM and Gregor. Accordingly, claim 9 is patentably distinguishable over Hammond, IBM, Gregor and Schacham at least because the claim includes all the limitations recited in parent

claims 1 and 8. The applied combination of Hammond, IBM, Gregor and Schacham would not have led an artisan to a modification that would result in the claimed invention.

2. Claim 11

Claim 11 is dependent from claim 1 and additionally requires the following:

> said translator translates the read out instruction in said second instruction architecture into the plurality of instructions as the instruction in said first instruction architecture, and said controller provides said first condition with one of said plurality of instructions and said second condition with each of said plurality of instruction but said one instruction.

The Examiner further applied Schacham for teaching the additional requirements in claim 11. However, Appellant submits that Schacham does not cure the deficiencies of the applied combination of Hammond, IBM and Gregor. Accordingly, claim 11 is patentably distinguishable over Hammond, IBM, Gregor and Schacham at least because the claim includes all the limitations recited in parent claims 1 and 8. The applied combination of Hammond, IBM, Gregor and Schacham would not have led an artisan to a modification that would result in the claimed invention.

G. Rejection of claim 14 under 35 U.S.C. §103(a) over Hammond, IBM and Mallick

Claim 14 is dependent from claim 12 and additionally requires the following:

> The instruction memory attached with a translator according to claim 12, further comprising an address translator to translate an address at the time of reading from said instruction storage unit.

The Examiner further applied Mallick for teaching the additional requirements in claim 14. However, Appellant submits that Schacham does not cure the deficiencies of the applied combination of Hammond and IBM. Accordingly, claim 14 is patentably distinguishable over Hammond and IBM at least because the claim includes all the limitations recited in parent claim 12. The applied

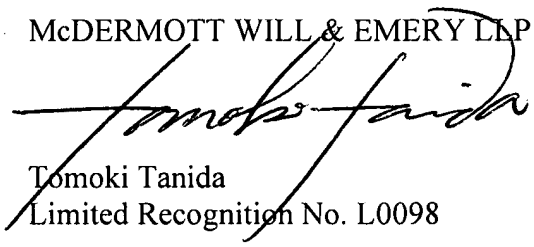
combination of Hammond, IBM and Mallick would not have led an artisan to a modification that would result in the claimed invention.

VIII. Conclusion

For all of the foregoing reason, Appellant respectfully submits that none of rejections of record is legally viable. Reversal of all rejections is respectfully solicited.

Respectfully submitted,

McDERMOTT WILL & EMERY LLP



Tomoki Tanida
Limited Recognition No. L0098

600 13th Street, N.W.
Washington, DC 20005-3096
Phone: 202.756.8000 SAB:TT:lnm
Facsimile: 202.756.8087
Date: **December 11, 2006**

Please recognize our Customer No. 20277 as our correspondence address.

IX. CLAIMS APPENDIX

1. In a processor operating with instructions in a first instruction architecture as a native instruction, an instruction translator used with an instruction memory to store an instruction in a second instruction architecture different from said first instruction architecture, for translating an instruction in said second instruction architecture into an instruction in said first instruction architecture for execution by said processor in said first instruction architecture said instruction translator comprising:

a translator for reading out an instruction from said instruction memory in response to a received first address in said instruction memory of an instruction to be executed by said processor and translating the read out instruction in said second instruction architecture into an instruction in said first instruction architecture, wherein said processor is configured to execute instructions only in said first instruction architecture;

an instruction cache for temporarily holding the instruction in said first instruction architecture after the translation by said translator in association with the first address in said instruction memory; and

a selector for searching said instruction cache in response to a received second address of an instruction to be executed by said processor, and for selectively outputting, based on a determination result of whether or not an instruction corresponding to the instruction of the second address is held in said instruction cache, an instruction output by said translator or the corresponding instruction held in said instruction cache.

2. The instruction translator according to claim 1, wherein said second instruction architecture is a variable length instruction architecture, and said translator includes a variable length translator for translating an instruction in said second instruction architecture read out from said instruction memory

into one or more instructions in said first instruction architecture, the number of which depends on an instruction length of the read out instruction in said second instruction architecture.

3. The instruction translator according to claim 2, wherein said variable length translator translates the instruction in said second instruction architecture read out from said instruction memory into an instruction in said first instruction architecture having a total length depending on and larger than the instruction length of said read out instruction in said second instruction architecture.

4. The instruction translator according to claim 3, wherein each instruction in said first instruction architecture includes one or a plurality of sub instructions, and the number of the sub instructions included in the instruction in the first instruction architecture translated by said variable length translator depends on the instruction length of said read out instruction in said second instruction architecture.

5. The instruction translator according to claim 1, wherein said translator includes a plurality of translators which translate a plurality of instructions in said second instruction architecture read out from said instruction memory into one instruction in said first instruction architecture.

6. The instruction translator according to claim 1, wherein each instruction in said first instruction architecture can include one or a plurality of sub instructions, and

said translator translates a plurality of instructions in said second instruction architecture read out from said instruction memory into an instruction in said first instruction architecture including sub instructions, the number of which depends on the number of said plurality of instructions.

7. The instruction translator according to claim 6, wherein the number of sub instructions included in the instruction in said first instruction architecture after said translation is equal to the number of said plurality of instructions.

8. The instruction translator according to claim 1, wherein

said translator translates said read out instruction in said second instruction architecture into one or a plurality of instructions as the instruction in said first instruction architecture;

said instruction translator further comprises a controller for controlling said instruction cache so that said instruction cache holds each of said one or said plurality of instructions held in said instruction cache as an entry which can be invalidated in one of first and second conditions.

9. The instruction translator according to claim 8, wherein said first condition is a holding control condition by hardware control based on a prescribed algorithm by said instruction cache, and

said second condition is a condition in which an explicit invalidation instruction is applied from the outside of said instruction cache.

10. The instruction translator according to claim 8, wherein said controller outputs a signal asserted when a new instruction cannot be held in said instruction cache without invalidating an entry which can be invalidated in the second condition.

11. The instruction translator according to claim 8, wherein

said translator translates the read out instruction in said second instruction architecture into the plurality of instructions as the instruction in said first instruction architecture, and said controller

provides said first condition with one of said plurality of instructions and said second condition with each of said plurality of instruction but said one instruction.

12. An instruction memory attached with a translator, used with a processor configured to execute instructions in a first instruction architecture as a native instruction, comprising:

an instruction storage unit to store an instruction in a second instruction architecture; and

an instruction translator to translate an instruction in said second instruction architecture output from said instruction storage unit into an instruction in said first instruction architecture for application to said processor, wherein said processor is configured to execute instructions only in said first instruction architecture;

said instruction translator including:

a translator to read out the instruction in said second instruction architecture from said instruction storage unit in response to a received first address of an instruction to be executed by said processor and translate the read out instruction in said second instruction architecture into the instruction in said first instruction architecture;

an instruction cache to temporarily hold the instruction in said first instruction architecture after the translation by said translator in association with the first address; and

a selector to search said instruction cache in response to a received second address of an instruction to be executed by said processor and selectively output to said processor, based on a determination result of whether or not an instruction corresponding to the instruction of the address is held in said instruction cache, an instruction output by said translator or the corresponding instruction in said first instruction architecture held in said instruction cache.

14. The instruction memory attached with a translator according to claim 12, further comprising an address translator to translate an address at the time of reading from said instruction storage unit.

16. A data processing apparatus, comprising:

a processor operating with an instruction in a first instruction architecture as a native instruction;

a bus to which said processor is connected;

a first instruction memory with a translator interconnected with said processor through said bus; and

a second instruction memory interconnected to said processor through said bus,

said first instruction memory with a translator including:

a first instruction storage unit to store an instruction in a second instruction architecture transferred from said processor through said bus; and

an instruction translator to translate the instruction in said second instruction architecture output from said first instruction storage unit into an instruction in said first instruction architecture for application to said processor through said bus, and

said second instruction memory including:

a second instruction storage unit to store an instruction in said first instruction architecture transferred from said processor through said bus; and

an instruction reading circuit responsive to an address signal applied from said processor through said bus for applying an instruction in said first instruction architecture output from said second instruction storage unit to said processor through said bus.

18. The data processing apparatus according to claim 16, wherein the number of clock cycles spent by said processor to access said first instruction memory through said bus is larger than the number of clock cycles spent by said processor to access said second instruction memory through said bus.

19. The data processing apparatus according to claim 16, further comprising a third instruction memory with a translator interconnected to said processor through said bus,

said third instruction memory with a translator including:

a third instruction storage unit to store an instruction in a third instruction architecture different from said second instruction architecture, said instruction in said third instruction architecture being transferred from said processor through said bus; and

an instruction translation circuit responsive to an address signal applied from said processor through said bus, for translating an instruction in said third instruction architecture output from said third instruction storage unit into an instruction in said first instruction architecture for application to said processor.

X. EVIDENCE APPENDIX

No evidence has been submitted of record under 37 CFR 1.130, 1.131 or 1.132.

XI. RELATED PROCEEDINGS APPENDIX

No decisions have been rendered in Related Appeals or Interferences.